LINGUAGEM DE MANIPULAÇÃO DE DADOS

Prof.: Geraldo Júnior

Atualmente, a SQL é a linguagem padrão adotada mundialmente em banco de dados. Ela é considerada um dos principais motivos para o sucesso comercial dos SGBD relacionais, com a utilização dessa linguagem foi possível substituir um SGBD por outro sem que todo o esforço de codificação tivesse que ser refeito.



TIPOS DE DADOS MAIS COMUNS:

- **SMALLINT: armazenar valores numéricos, em dois bytes binários, compreendidos no intervalo de -32768 a +32767.**
- **INTEGER: armazenar valores numéricos, em quatro bytes binários, compreendidos no intervalo de -2147483648 a +2147483647.**
- **DECIMAL (n,m): armazenar valores numéricos com, no máximo, 15 dígitos. Nesta opção deve ser definida a quantidade de dígitos inteiros (n) e casas decimais (m) existentes no campo.**
- **BOOLEAN: armazenar valores booleanos ou lógicos, tais como TRUE/FALSE.**

TIPOS DE DADOS MAIS COMUNS:

- **BIT:** armazenar os valores lógicos 0 ou 1.
- **FLOAT:** armazena valores numéricos aproximados com precisão de ponto flutuante.
- **DATETIME:** armazena data com 8 dígitos e a hora com precisão centésimos de segundos.
- **DATE:** armazena data com 8 dígitos.
- **TIME:** A hora é armazenada com precisão até centésimos de segundos.
- **CHAR:** armazenar um único caractere alfanumérico.
- **CHAR (n):** armazenar n caracteres alfanuméricos, onde n deve assumir um valor entre 1 e 254 caracteres.

TIPOS DE DADOS MAIS COMUNS:

- **VARCHAR (n):** armazenar até n caracteres alfanuméricos, onde n deve assumir um valor entre 1 e 254 caracteres. A diferença em relação ao tipo anterior, é que neste caso, o registro não terá um tamanho fixo de caracteres, variando de acordo com a necessidade em cada caso, desde que não ultrapasse o tamanho máximo (254 caracteres), dessa forma, é mais econômico em termos de memória, no entanto, exige maior esforço do desenvolvedor.

- **LONG VARCHAR:** armazenar acima de 254 caracteres alfanuméricos.

STURCTURED QUERY LANGUAGE (SQL)

A linguagem SQL se tornou padrão para ambientes de banco de dados relacionais, quando o American National Standard Institute (ANSI) reconheceu a importância da linguagem e definiu normas para sua padronização e segue fazendo atualizações regularmente.

A SQL possui recursos de álgebra relacional e de cálculo relacional, mesmo com uma sintaxe de fácil compreensão. Entre as muitas facilidades, ela dispõe de propriedades que permitem a inclusão de instruções dentro do código de linguagens de programação de uso geral, tais como: COBOL, C/C++Java e Python, entre outras.

STURCTURED QUERY LANGUAGE (SQL)

A SQL se divide nas seguintes partes:

- **Linguagem de Definição de Dados (DDL): disponibiliza comandos para definir e descrever tabelas, visões, índices e outros objetos do banco de dados. Além disso, pode dispõe de comandos para especificação de restrições de integridade e de permissões de acesso (DCL);**
- **Linguagem de Manipulação de Dados (DML): disponibiliza comando para consulta, inserção, exclusão e modificação de registros no banco de dados;**
- **Controle de transações: comandos para acompanhamento de início e fim das transações.**

DDL – LINGUAGEM DE DEFINIÇÃO DE DADOS (DATA DEFINITION LANGUAGE)

Comando:	**CREATE DATABASE**
Funcionalidade:	Efetua a criação de um banco de dados dentro de um SGBD disponível
Sintaxe:	`CREATE DATABASE nome_do_banco_de_dados`
Observações:	Na maioria dos SGBD comerciais será preciso ter autorização para a utilização desse comando. Esses comandos fazem parte da DCL e serão vistos mais à frente.

Comando:	**DROP DATABASE**
Funcionalidade:	Executa a remoção completa de um banco de dados existente dentro de um SGBD disponível, e toda a sua estrutura incluindo tabelas e dados nelas impostados.
Sintaxe:	`DROP DATABASE nome_do_banco_de_dados`
Observações:	Na maioria dos SGBD comerciais será preciso ter autorização para a utilização desse comando. Esses comandos fazem parte da DCL e serão vistos mais à frente.

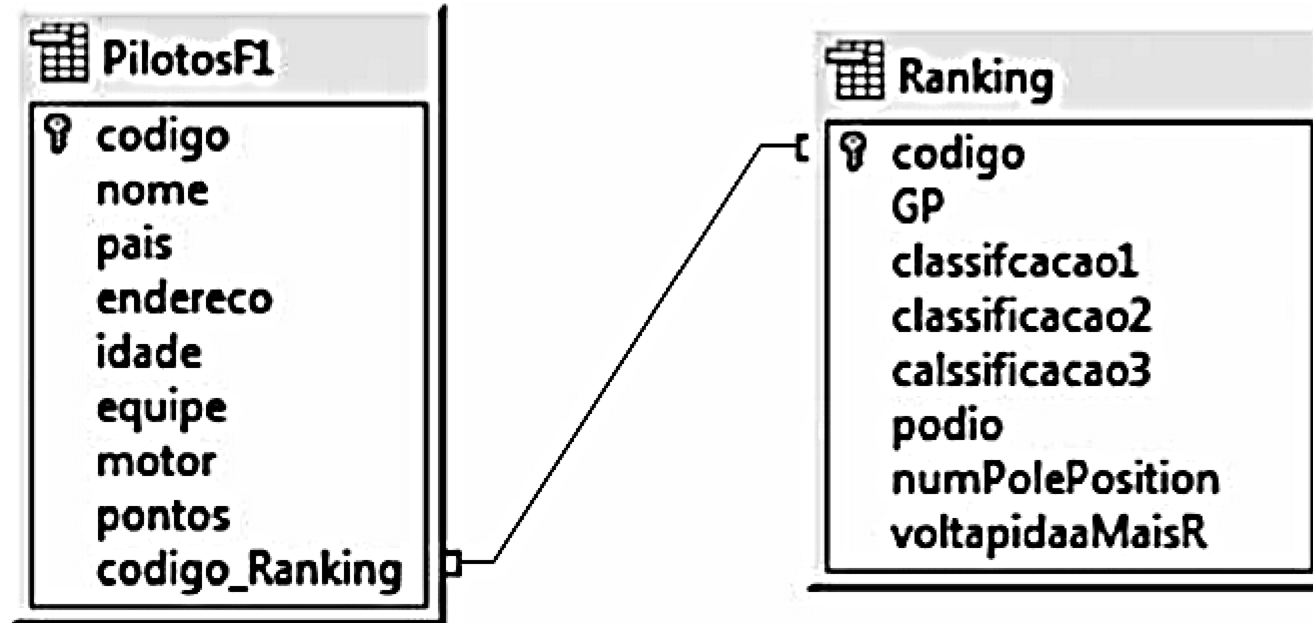
Comando:	**CREATE TABLE**
Funcionalidade:	Efetua a criação de uma tabela dentro do banco de dados
Sintaxe:	CREATE TABLE nome_da_tabela(atributo1 Tipo, atributo2 Tipo, atributon Tipo)
Observações:	Cada coluna (atributo) tem que ter um nome e um tipo de dado associado a ele.

Na criação de uma tabela, devemos definir um atributo como chave primária (PK – Primary Key) ou como chave estrangeira (FK – Foreing Key). Existem duas sintaxes possíveis:

```
CREATE TABLE nome_da_tabela(
             atributo1 Tipo,
             atributo2 Tipo,
             atributox(PK) Tipo,
             atributoy(FK) Tipo,
             atributon Tipo)
```

```
CREATE TABLE nome_da_tabela(
             atributo1 Tipo,
             atributo2 Tipo,
             atributon Tipo,
             atributoz Tipo
             PRIMARY KEY (atributo1)
             FOREIGN KEY (atributoz)
             )
```

Exemplo de chaves primárias e chave estrangeira



ALTER TABLE PilotosF1
ADD codigo_Ranking int(11);

ALTER TABLE PilotosF1
ADD FOREIGN KEY(codigo_Ranking)
REFERENCES Ranking(codigo);

Comando:	**DROP TABLE**
Funcionalidade:	Executa a remoção completa de uma tabela existente no banco de dados e toda a sua estrutura incluindo os dados impostados nela.
Sintaxe:	`DROP TABLE nome_da_tabela`
Observações:	Na maioria dos SGBD comerciais será preciso ter autorização para a utilização desse comando. Esses comandos fazem parte da DCL e serão vistos mais à frente.

Comando:	**ALTER TABLE**
Funcionalidade:	Efetua a alteração na estrutura de uma tabela dentro do banco de dados, de acordo com a necessidade pode adicionar uma nova coluna [ADD], modificar uma coluna existente [MODIFY] ou eliminar uma coluna existente [DROP].

Sintaxe:	`ALTER TABLE nome_da_tabela(`
	`[ADD] nome_coluna Tipo_coluna,`
	`[MODIFY] nome_coluna Tipo_coluna,`
	`[DROP] nome_coluna Tipo_coluna,`
	`)`

Observações:	Cada coluna (atributo) tem que ter um nome e um tipo de dado associado a ele.

DML – LINGUAGEM DE MANIPULAÇÃO DE DADOS (DATA MANIPULATION LANGUAGE)

Comando:	**SELECT**
Funcionalidade:	Selecionar registros em uma ou mais tabelas
Sintaxe:	SELECT [*/ALL/DISTINCT] atributo1, atributo2, atributo*n*] FROM nome_da_tabela [, nome_da_tabela2] WHERE <condição> GROUP BY <nome-atributo> [,<nome-atributo2>] [HAVING<condição>] ORDER BY <nome-atributo> [,<nome-atributo2>] [ASC/DESC]
Observações:	

Operações Matemáticas que podem ser utilizadas com o comando SELECT:

Operação:	**COUNT**
Funcionalidade:	Retorna a quantidade de registros encontrados no campo especificado na consulta.
Sintaxe:	SELECT COUNT [(*/ALL/DISTINCT] atributo1, atributo2, atributo*n*)]
Observações	Quando a opção * (sinal asterisco) é utilizada, o resultado é o total de registros existentes. Quando é referenciado o nome de uma coluna (atributo) retorna a quantidade de valores existentes na coluna.

Operação:	**SUM**
Funcionalidade:	Retorna a soma dos valores existentes na coluna (atributo) especificada.
Sintaxe:	SELECT SUM ([*/ALL/DISTINCT] <nome_atributo>)
Observações	Quando a opção * (sinal asterisco) é utilizada, o resultado é o total de registros existentes. Quando é referenciado o nome de uma coluna (atributo) retorna a quantidade de valores existentes na coluna. A coluna (atributo) deve ser de um tipo numérico, conforme definido na criação da tabela.

Operações Matemáticas que podem ser utilizadas com o comando SELECT:

Operação:	**AVG**
Funcionalidade:	Retorna a média aritmética dos valores existentes na coluna (atributo) especificada.
Sintaxe:	SELECT AVG ([*/ALL/DISTINCT] <nome_atributo>)
Observações	Quando a opção * (sinal asterisco) é utilizada, o resultado é o total de registros existentes. Quando é referenciado o nome de uma coluna (atributo) retorna a quantidade de valores existentes na coluna. A coluna (atributo) deve ser de um tipo numérico, conforme definido na criação da tabela.

Operação:	**MAX**
Funcionalidade:	Retorna o maior valor encontrado na coluna (atributo) na coluna especificada.
Sintaxe:	SELECT MAX ([*/ALL/DISTINCT] <nome_atributo>)
Observações	Quando a opção * (sinal asterisco) é utilizada, o resultado é o total de registros existentes. Quando é referenciado o nome de uma coluna (atributo) retorna a quantidade de valores existentes na coluna. A coluna (atributo) deve ser de um tipo numérico, conforme definido na criação da tabela.

Operações Matemáticas que podem ser utilizadas com o comando SELECT:

Operação:	**MIN**
Funcionalidade:	Retorna o menor valor encontrado na coluna (atributo) na coluna especificada.
Sintaxe:	SELECT MIN ([*/ALL/DISTINCT] <nome_atributo>)
Observações	Quando a opção * (sinal asterisco) é utilizada, o resultado é o total de registros existentes. Quando é referenciado o nome de uma coluna (atributo) retorna a quantidade de valores existentes na coluna. A coluna (atributo) deve ser de um tipo numérico, conforme definido na criação da tabela.

DML – LINGUAGEM DE MANIPULAÇÃO DE DADOS (DATA MANIPULATION LANGUAGE)

Comando:	**INSERT**
Funcionalidade:	Incluir um novo registro em uma tabela do banco de dados
Sintaxe:	INSERT INTO <nome_da_tabela> [<nome_atributo1>, <nome_atributo2, nome_atributo*n*] VALUES (<valor> [<valor_atributo1>, <valor_atributo2, valor_atributo*n*]
Observações:	Os valores a serem inseridos devem ser iguais àqueles existentes na definição de cada coluna (atributo) da tabela.

INSERT INTO Funcionarios (CPF, nome, endereco, telefone, salario, funcao)
VALUES (30361290876, 'Ademir José Silva', 'Campinas', 22317865, 0, NULL);

Podemos executar a consulta:

SELECT * FROM Funcionarios;

Como resultado da consulta, teremos a seguinte tabela:

CPF	nome	endereco	telefone	salario	funcao
30361290876	Ademir José Silva	Campinas	22317865	0.00	NULL

DML – LINGUAGEM DE MANIPULAÇÃO DE DADOS (DATA MANIPULATION LANGUAGE)

Comando:	**UPDATE**
Funcionalidade:	Promove a alteração no(s) valor(es) de um ou mais colunas de uma tabela existente no banco de dados
Sintaxe:	UPDATE nome_da_tabela SET <nome_atributo1> =<novo_ valor_atributo1>, [<nome_atributo2> =<novo_ valor_atributo2>, <nome_atributo*n*> =<novo_ valor_atributo*n*>] WHERE <condição>
Observações:	Esta atualização poderá ocorrer em um, vários ou até mesmo sobre todos os registros de uma tabela.

```
UPDATE Funcionarios
SET funcao= 'Auxiliar',
salario = 800
WHERE CPF=32361298734;

SELECT * FROM Funcionarios;
```

CPF	nome	endereco	telefone	salario	funcao
20321295096	João Alberto Smith	Itatiba	22447865	0.00	NULL
23161197770	Ana Salles Azir	Salto	22317865	0.00	Recepcionista
30361290876	Ademir José Silva´	Campinas	22317865	0.00	NULL
32361298734	Luís Henrique Talles	Campinas	21531785	800.00	Auxiliar
45403612087	Francisco José Almeida	Indaiatuba	25417761	0.00	NULL
61254590871	Lucia Vicentim	Salto	21316565	0.00	Bibliotecaria

DML – LINGUAGEM DE MANIPULAÇÃO DE DADOS (DATA MANIPULATION LANGUAGE)

Comando:	**DELETE**
Funcionalidade:	Remove um ou mais registros de uma tabela do banco de dados
Sintaxe:	DELETE [*/ALL/DISTINCT] atributo1, atributo2, atributo*n*] FROM nome_da_tabela WHERE <condição>
Observações:	Esta atualização poderá ocorrer em um, vários ou até mesmo sobre todos os registros de uma tabela.

DML – LINGUAGEM DE MANIPULAÇÃO DE DADOS (DATA MANIPULATION LANGUAGE)

CPF	nome	endereco	telefone
10122020232	Maria de Lourdes Amaral	NULL	35440089
19321122213	José Francisco de Paula	NULL	27219756
22539910976	Ivete Medina Chernell	NULL	48170352
45399109881	Raquel Santos	NULL	87603451
70964411900	Luiza Souza Prado	NULL	34559087

```
DELETE FROM Usuarios
WHERE CPF = 45399109881;
SELECT * FROM Usuarios;
```

CPF	nome	endereco	telefone
10122020232	Maria de Lourdes Amaral	NULL	35440089
19321122213	José Francisco de Paula	NULL	27219756
22539910976	Ivete Medina Chernell	NULL	48170352
70964411900	Luiza Souza Prado	NULL	34559087

OBRIGADO!

ATÉ A PRÓXIMA AULA.